Software II - Advanced Java Concepts – C195

READ ME File

1. **Create a log-in form that can determine the user’s location and translate log-in and error control messages (e.g., “The username and password did not match.”) into two languages.**

Checks Database if Username and Password Don’t match:

* + Login Info
    - UserName: test
    - Password: test
    - login() User.java line 44
      * Checks Username and Password from database.

Checks location and set Language:

* + setLanguage() – LoginController.java Line58
    - get Local.getDefault()); default English if “no -Norwegian” or “en- English” and then get strings from Languages/lang\*

1. **Provide the ability to add, update, and delete customer records in the database, including name, address, and phone number.**

Add:

* + AddCustomerSave() – View\_Controller/AddCustomerController.java line 86 (Gets Text from fields)
    - saveCustomer() – Model/Customer.java line 101 (Saves to Database)

Update:

* + UpdateCustomerSave() – View\_Controller/UpdateCustomerController.java line 93 (Gets Text from fields)
    - updateCustomer() – Model/Customer.java line 126 (Saves to Database)

Delete:

* + DeleteCustomer() – View\_Controller/MainController.java line 237 (Gets Text from fields)
    - deleteCustomer() – Model/Customer.java line 150 (Saves to Database)

1. **Provide the ability to add, update, and delete appointments, capturing the type of appointment and a link to the specific customer record in the database.**

Add:

* newAptSave() – View\_Controller/AddAptController.java – line 157 (Gets Text from fields)
  + addApt() – Model/Appointment.java – line 411 (Saves to Database)

Update:

* updateAptSave() – View\_Controller/UpdateAptController.java – line 177 (Gets Text from fields)
  + updateApt() – Model/Appointment.java – line 431 (Saves to Database)

Delete:

* DeleteApt() – View\_Controller/MainController.java – line 348 (Gets Text from fields)
  + deleteapt() – Model/Appointment.java – line 451 (Saves to Database)

1. **Provide the ability to view the calendar by month and by week**.

All:

* AllAptTable - View\_Controller/MainController.java
  + getAllAppointments() – Model/Appointment- line 153 (Gets all Appointment for User to be displayed in table)

Month:

* MonthAptTable - View\_Controller/MainController.java
  + getMonthAppointments() – Model/Appointment- line 205 (Gets all Appointment for User that month to be displayed in table)

Week:

* AllAptTable - View\_Controller/MainController.java
  + getWeekAppointments() – Model/Appointment- line 254 (Gets all Appointment for User that week to be displayed in table)

1. **Provide the ability to automatically adjust appointment times based on user time zones and daylight saving time.**

All functions use “ZoneId.of” to switch LocalDateTime so they are correctly changed with correct time zone and daylight saving time

Switches UTC to Location

* getLocationDateTime() – Model/Appointment – line 312

Switches Location to Appointment Time

* getAptLocationDateTime() – Model/Appointment – line 351

Puts Time and Datetime together and switch from apt location to UTC

* getDateTime() – Model/Appointment – line 399

F.   **Write exception controls to prevent each of the following. You may use the same mechanism of exception control more than once, but you must incorporate at least  two different mechanisms of exception control.**

* scheduling an appointment outside business hours
* When adding or updating Appointment
  + Date: On DatePicker, Datacell Saturday and Sunday are disabled.
  + Time: AptStartTime and AptEndTime observableArrayList doesn’t list any times that are outside business hours
* scheduling overlapping appointments
  + appointmentAvialableuser() – Model/Appointment – line 467
    - Check if Apt time conflicts with any other apt for user
  + appointmentAvialableCust() – Model/Appointment – line 486
    - Check if Apt time conflicts with any other apt for customer
* entering nonexistent or invalid customer data
* isClientValid() – Model/Customer.java – line 175
  + Checks if any required filled is left blank or invalid for customer
* isAptValid() – Model/Customer.java – line 503
  + Checks if any required filled is left blank or invalid for appointment
* entering an incorrect username and password
* login() – Model/User.java – line 44
  + Check if Username and Password is in database. If false it will display error message

1. **Write two or more lambda expressions to make your program more efficient, justifying the use of each lambda expression with an in-line comment.**

First Lambda:

* Model/Appointment.java – line 189, line 238, line 287
  + Gets all appointment in Observable list and change from UTC time to Users LocalDateTime.

Second Lambda:

* View\_Controller/MainController.java – line 309
  + Combines all Three TableView (AllApt,MonthApt,WeekApt) and will get selected from any table to pull information into UpdateAptController.

1. **Write code to provide an alert if there is an appointment within 15 minutes of the user’s log-in.**

* checkForAppt() – View\_Controller/MainController.java – line 407
  + initialize on MainController after log-in
  + Checks users localDate Time against database within 15 min

1. **Provide the ability to generate each  of the following reports:**

* number of appointment types by month
  + reportMonth() -View\_Controller/ReportController.java – line 81
    - the schedule for each consultant
* reportSchedule() -View\_Controller/ReportController.java – line 52
  + get userID, username, Appointment ID, Type, Start, and End (Displayed in UTC Time)
    - one additional report of your choice
* reportSLocation() -View\_Controller/ReportController.java – line 105
  + get amount of appointment for location for month

**J.**   **Provide the ability to track user activity by recording timestamps for user log-ins in a .txt file. Each new record should be appended to the log file, if the file already exists.**

Log file in Support Doc/logins.txt

* Login()- Model/Login.java – line 16
  + Saves/Create Support Doc/login.txt
  + Writes to login.txt on login with success or failure

1. **Demonstrate professional communication in the content and presentation of your submission.**

* Provided “Readme” File with how project requirement are meant with function locations.
* List inline comments throughout the program